Bab 3

Mengarahkan Flow Program, Input, dan Pemrosesannya

Di modul ini, Anda akan belajar beberapa teknik pemrograman dan pengetahuan penting yang perlu diketahui karena sering dijumpai dalam memprogram.

3.1 Mengatur Alur Pemrograman

Sebuah program diakses secara top-down, dari atas ke bawah. Jika tidak adanya pengaturan alur, tentu program akan tidak bisa "berbuat banyak" dan hanya mengeksekusi kode dari atas ke bawah saja. Untuk itu Anda bisa melakukan pengaturan alur pemrograman menggunakan beberapa command, seperti IF statement dan Switch.

3.1.1 IF Statement

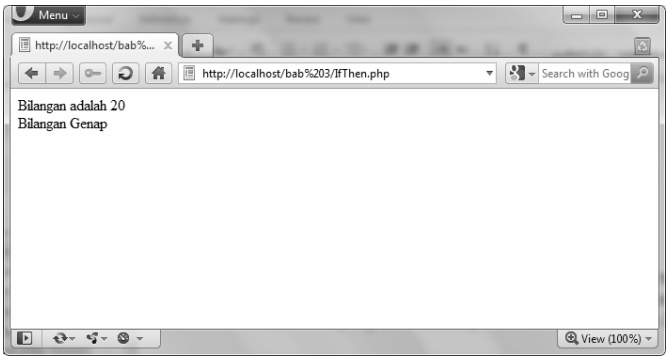
Statement IF merupakan statement yang penting dan pasti terdapat di semua bahasa pemrograman. Statement ini berguna untuk membuat percabangan berdasarkan kondisi tertentu yang harus dipenuhi.

```
If (kondisi tertentu = True) {
aksi
}
```

Contoh kode untuk program yang menggunakan IF statement di PHP adalah:

```
<?php
$bilangan = 20;
print "Bilangan adalah ".$bilangan;
print "</br>";
if ( $bilangan%2 == 0 ) {
print "Bilangan Genap";
}
?>
```

Kode di atas akan menampilkan teks “Bilangan Genap” karena angka = 20 merupakan bilangan genap. Operator “==” digunakan untuk membandingkan statement yang hasil akhirnya adalah nilai True atau False.



Gambar 3.1 If then untuk menentukan bilangan genap atau tidak

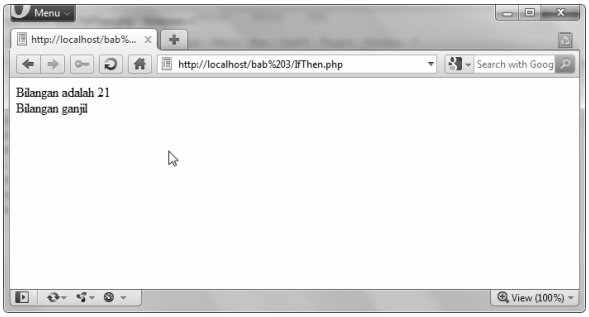
Kode IF statement di atas hanya menampilkan satu alternatif solusi, sehingga jika kondisi IF tidak dipenuhi, tidak ada aksi apa pun dan blok kode aksi akan dilewati. Untuk membuat alternatif terdiri atas 2 buah, Anda bisa menggunakan IF-ELSE dengan cara penulisan seperti berikut:

```
if (kondisitertentu = True) {
Aksi jika kondisi True
} else {
Aksi jika kondisi False
}
```

Contoh penggunaan kode IF-ELSE terlihat seperti berikut:

```
<?php
$bilangan = 21;
print "Bilangan adalah ".$bilangan;
print "</br>";
if ( $bilangan%2 == 0 ) {
print "Bilangan Genap";
} else {
print "Bilangan ganjil";
}
?>
```

Kode di atas akan menampilkan “Bilangan Ganjil” di layar karena nilai $bilangan = 21 merupakan bilangan ganjil.



Gambar 3.2 Tampilan saat bilangan = 21

3.1.2 Switch Statement

Statement untuk pengaturan alur program berikutnya adalah switch. Salah satu keuntungan switch adalah Anda bisa langsung mengevaluasi satu statement dan memerintahkan aksi dalam jumlah yang lebih banyak. Statement switch memiliki sintaks seperti berikut:

```
switch (ekspresi) {
    case hasil_1:
      aksi_1;
      break;
    case hasil_2:
      aksi_2;
      break;
    default:
        aksi jika tidak ada case yang cocok;
}
```

Untuk lebih jelasnya, silakan lihat contoh kode berikut:

```
<?php
$nilai_ujian = 90;
print "Nilai Anda ".$nilai_ujian;
print "<br/>";
switch ($nilai_ujian) {
case ($nilai_ujian<50):
     print "Maaf, tidak lulus";
     break;
case ($nilai_ujian>=50):
     print "Anda lulus";
     break;
default:
     break;
}
?>
```

Jika dieksekusi, kode di atas akan menghasilkan tulisan seperti gambar di bawah ini.

Gambar 3.3 Switch digunakan untuk melakukan switching guna mengatur flow program

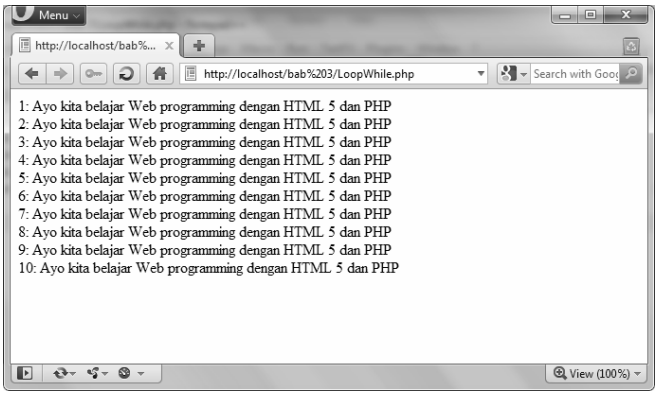
Kelebihan switch adalah alternatif tindakan yang diambil bisa langsung banyak tanpa harus menulis IF-ELSE secara bertingkat yang lebih sulit dibaca. Jangan lupa untuk menuliskan break jika Anda ingin keluar dari switch ketika aksi sudah dilakukan.

3.1.3 Loop While

Perbedaan komputer dengan manusia adalah komputer mampu disuruh untuk melakukan pekerjaan yang sama berulang-ulang tanpa pernah merasa capek. Untuk memberikan perintah ke komputer agar melakukan kegiatan yang berulang-ulang, Anda bisa menggunakan Loop.

```
<?php
$jumlah_ulangan = 0;
while ( $jumlah_ulangan < 10 ) {
$jumlah_ulangan++;
print $jumlah_ulangan.": Ayo kita belajar Web programming
dengan HTML 5 dan PHP <br />"; }
?>
```

Jika dibuka, loop while di atas menghasilkan repetisi seperti gambar berikut.

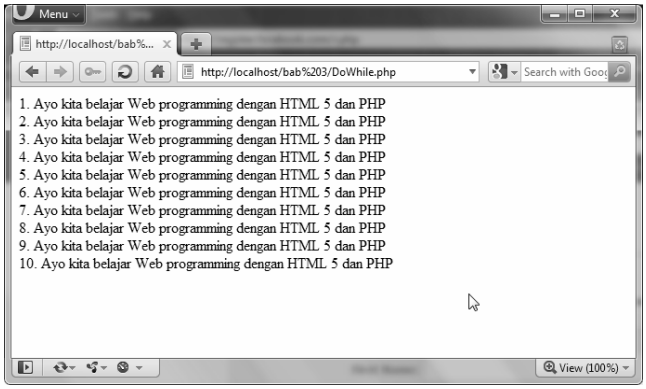


Gambar 3.4 Loop While untuk menghasilkan pengulangan

Do While merupakan teknik lain untuk membuat loop, hanya saja statement kondisional diletakkan di bagian bawah. Jika kode di atas diubah menjadi Do While, akan menjadi seperti berikut:

```
<?php
$jumlah_ulangan = 1;
do {
print $jumlah_ulangan.". Ayo kita belajar Web programming
dengan HTML 5 dan PHP <br />";
$jumlah_ulangan++;
} while (  $jumlah_ulangan <= 10);
?>
```

Tampilannya seperti berikut ini.



Gambar 3.5 Do while untuk melakukan repetisi

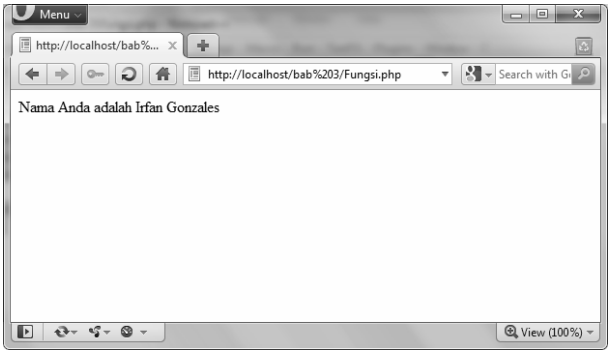
3.2 Membuat Fungsi

Di pemrograman juga ada istilah fungsi/function. Fungsi membuat proses pemrograman bisa lebih efisien karena aksi-aksi tertentu bisa ditulis dalam satu kali fungsi yang nantinya bisa dipanggil jika diperlukan.

Sebuah fungsi dapat menerima nilai, memprosesnya kemudian mengembalikan nilai barunya. Secara fisik, fungsi merupakan blok kode yang bisa dipanggil nantinya.

```
<?php
function sebutkan( $nama) {
 $hasil = "Nama Anda adalah ".$nama;
 Print $hasil;
}
print sebutkan("Irfan Gonzales");
?>
```

Jika dijalankan di browser, tampilannya menjadi seperti berikut ini.



Gambar 3.6 Penampilan nama dengan fungsi

Dalam fungsi di atas terdapat variabel. Variabel yang diletakkan di dalam fungsi cakupannya lokal, artinya tidak dapat diakses di luar fungsi tersebut.

Di dalam fungsi, Anda juga tidak dapat mengakses fungsi lain di luarnya.

```
<?php
$nama = "Rafael Nadal";
function sebutkan() {
     print "Nama Anda adalah $nama <br />";
}
sebutkan();
?>
```

Jika kode di atas dieksekusi, maka tidak akan ditampilkan $nama mengingat variabel tersebut dideklarasikan di luar fungsi.

Gambar 3.7 Variabel yang diletakkan di luar fungsi tidak akan terlihat

3.3 Membuat Array

Sebuah bahasa pemrograman umumnya memiliki variabel berjenis array. Array (dalam bahasa Indonesia kerap diterjemahkan sebagai larik) merupakan variabel yang di dalamnya mengandung nilai-nilai lainnya.

Untuk mengalokasikan nilai array ke variabel tertentu, Anda dapat menggunakan banyak metode, salah satunya seperti berikut:

```
$manusia = array (
      "nama" => "Stephanie",
      "umur" => 25,
      );
```

Untuk mengakses, Anda bisa menggunakan nomor atau nama indeks dari nilai yang dimaksud, misalnya:

```
print $manusia['nama];
```

Array sendiri punya banyak variasi untuk mendefinisikan. Pertama dengan mengalokasikan indeks secara otomatis (di mana indeks pertama adalah angka 0) dengan kode berikut ini:

```
$mobil=array("Proton","Daihatsu","Citroen","Toyota");
```

Adapun cara yang kedua adalah dengan mengalokasikan indeks secara manual:

```
$mobil[0]="Proton";
$mobil[1]="Daihatsu";
$mobil[2]="Citroen";
$mobil[3]="Toyota";
```

Berikut ini contoh kode mendefinisikan array, lalu mengakses nilai array tersebut.

```
<?php
$makanan[0]="Pecel";
$makanan[1]="Rujak";
$makanan[2]="Petis";
$makanan[3]="Sate";
$makanan[4]="Rendang";
echo "Saya suka makan ".$makanan[1] . " dan " . $makanan[3] .
". Semuanya mak nyuss.";
?>
```

Jika dibuka di browser, array di atas menjadi seperti berikut.

Gambar 3.8 Halaman yang mengakses kode array di halaman

3.4 Parsing Variabel Antar Halaman

Sebuah halaman web sering kali memproses nilai yang diperoleh oleh halaman lainnya. Untuk itu PHP harus mampu mem-parsing variabel antar halaman. Ada banyak cara untuk mem-parsing nilai antar variabel, salah satunya adalah via URL.

Jika di PHP sebelumnya Anda bisa memanggil variabel dari halaman lain dengan langsung menggunakan nama variabel, kini Anda perlu menuliskannya dengan menambahkan sintaks khusus.

Kali ini Anda akan mencoba mem-parsing nilai antar variabel menggunakan sintaks $_REQUEST['variabel'], sementara di modul berikutnya Anda bisa melihat parsing yang lebih advance dengan form menggunakan $_POST. Buatlah file halaman1.php dengan kode seperti berikut:

```
<html>
<head>
<?php
$nama="BambangPamungkas";
print "<title>";
print "Parsing nama ".$nama;
print "</title>";
?>
<body>
<h1> Demo Parsing Variabel Antar Halaman</h1>
Nilai yang akan diparsing: <br/>
<b> nama </b> = <b><?php print $nama?></b>
<br/>
<?php
print "<noscript>";
```

```
print "Tolong aktifkan JavaScript";
print "</noscript>";
print "<button
onClick=window.location='halaman2.php?nama=$nama'>Coba Klik
Untuk menampilkan nama di Halaman kedua !</button>";
print "</a>"
?>
</body>
</head>
</html>
```

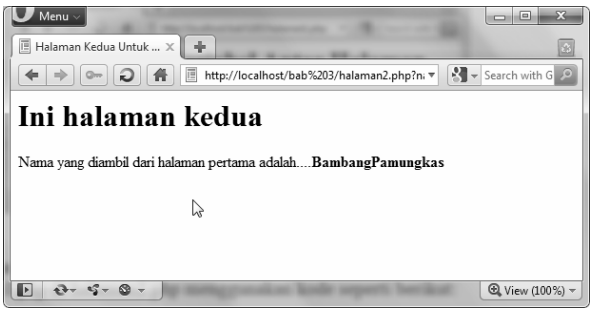
Jika dibuka akan terlihat seperti berikut, klik tombol yang ada, tapi sebelumnya buat dulu halaman2.php dengan kode seperti berikut.

Gambar 3.9 Tampilan halaman1.php ketika belum ditampilkan

Kemudian buat kode kedua yang akan menampilkan nilai yang di-parsing dari halaman1.php di halaman2.php menggunakan kode seperti berikut:

```
<html>
<head>
<title>
<?php
print "Halaman Kedua Untuk Menampilkan nama
".$_REQUEST['nama'];
?>
</title>
<body>
<h1> Ini halaman kedua</h1>
<?php
$nama= $_REQUEST['nama'];
print "Nama yang diambil dari halaman pertama
adalah....<b>".$nama."</b>";
?>
</body>
</head>
</html>
```

Jika tombol di halaman1.php diklik, muncul halaman2.php seperti berikut ini.



Gambar 3.10 Tampilan halaman2.php

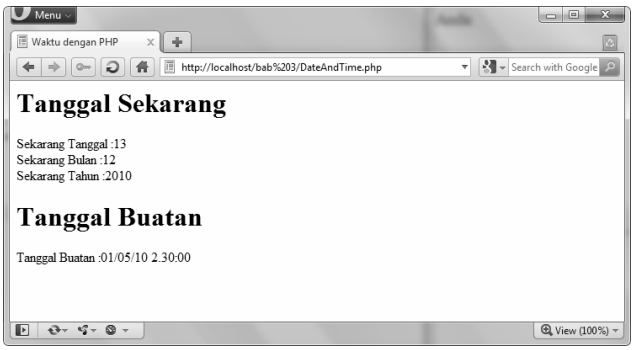
3.5 Bekerja dengan Date and Time

Dalam pemrograman juga penting adanya waktu. Di PHP, waktu sekarang bisa diperoleh dengan perintah getdate(); yang bisa ditampung di variabel yang bertipe array.

Sementara untuk mengalokasikan nilai tanggal tertentu ke variabel, Anda bisa menggunakan perintah mktime, lihat contohnya di kode berikut:

```
<html>
<head>
<title>Waktu dengan PHP</title>
</head>
<body>
<div>
<?php
echo "<h1>Tanggal Sekarang</h1>";
$tgl_Sekarang=getdate();
print "Sekarang Tanggal
:".$tgl_Sekarang['mday']."<br/>"."Sekarang Bulan
:".$tgl_Sekarang['mon']."<br/>"."Sekarang Tahun
:".$tgl_Sekarang['year']."<br />";
echo "<h1>Tanggal Buatan</h1>";
$waktu_buatan = mktime( 2, 30, 0, 5, 1, 2010 );
print "Tanggal Buatan :".date("d/m/y G.i:s", $waktu_buatan);
?>
</div>
</body>
</html>
```

Kode di atas, jika diperlihatkan di browser akan terlihat seperti berikut ini.



Gambar 3.11 Tampilan Date and Time

3.6 Tracking Pengakses

PHP memiliki kemampuan untuk mendeteksi browser pengakses halaman web. Hal ini memungkinkan Anda membuat halaman yang bisa men-tracking pengunjungnya. Selain itu PHP juga mampu bekerja dengan form yang memungkinkan halaman web mengumpulkan data yang digunakan oleh pengunjung. Tracking dan Form berkaitan erat karena sama-sama menggunakan variabel register global.

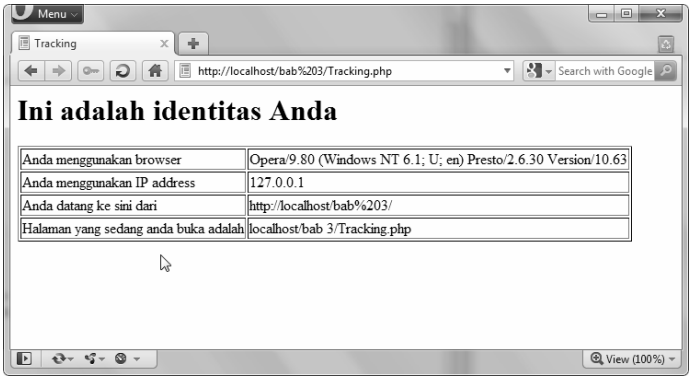
Tracking pengakses digunakan untuk mencari tahu identitas pengakses. Jika Anda penah menggunakan layanan extreme-tracking, sitestat, atau statcheck, maka tracking akan memberikan informasi-informasi dasar tentang identitas pengakses.

Buatlah kode berikut ini untuk membuat halaman web yang berisi tracking kemudian simpan dalam format .php.

```
<html>
<head> <title>
Tracking
</title> </head>
<body>
<h1> Ini adalah identitas Anda </h1>
<table border="1">
<tr>
```

```
<td>Anda menggunakan browser </td>
<td><?php print $_SERVER['HTTP_USER_AGENT'] ?> </td>
</tr>
<tr>
<td> Anda menggunakan IP address </td>
<td> <?php print $_SERVER['REMOTE_ADDR'] ?></td>
</tr>
<tr>
<td> Anda datang ke sini dari </td>
<td> <?php
  if ( ! empty( $_SERVER['HTTP_REFERER'] ) ) {
   print $_SERVER['HTTP_REFERER']; }
   else {
 print "Tidak ada yang membawa anda kesini";
  }
  ?></td>
</tr>
<tr>
<td> Halaman yang sedang anda buka adalah </td>
<td> <?php print $_SERVER['HTTP_HOST'].$_SERVER['PHP_SELF']
?></td>
</tr>
</table>
</body>
</html>
```

Ketika diakses di browser, Anda bisa melihat identitas Anda terpampang di server, yaitu Jenis browser, ip address, referer, dan halaman yang dibuka.



Gambar 3.12 Tampilan tracking pengakses dengan kode PHP

Dari kode di atas, Anda bisa mengetahui bahwa PHP mampu melakukan tracking menggunakan variabel global $_SERVER yang akan menampilkan data dari server. Variabel $_SERVER ini

bertipe array di mana item-item di dalamnya bisa digunakan untuk tracking, seperti:

- $_SERVER['HTTP_USER_AGENT']: Untuk mengetahui jenis browser yang digunakan.
- $_SERVER['REMOTE_ADDR']: Untuk mengetahui alamat IP pengakses.
- $_SERVER['HTTP_REFERER']: Untuk mengetahui referer (halaman yang digunakan untuk sampai ke halaman ini).
- $_SERVER['HTTP_HOST']: Untuk mengetahui alamat host server.
- $_SERVER['PHP_SELF']: Untuk mengetahui file PHP yang sedang dibuka.

Untuk menghasilkan layout yang baik, tampilan di atas dikurung di dalam tabel sehingga menghasilkan tampilan yang bagus.

3.7 Bekerja dengan Form

Di subbab sebelumnya telah dijelaskan tentang parsing variabel antar halaman. Sekarang Anda akan menerapkannya dengan lebih advance, yaitu menggunakan form. Form memungkinkan Anda menerima input secara aktif dari user tanpa harus hard coding di halaman web.

3.7.1 Form dengan Input dan Pemrosesan Terpisah

Form di HTML 5 dikenal dengan adanya tag <FORM> dan ditutup dengan tag </FORM>. Di dalam tag pembuka <FORM> diikuti dengan atribut action dan method. Action menjelaskan ke halaman yang digunakan untuk memproses input, sementara method digunakan untuk mengatur cara mem-parsing konten. Ada 2 method yang biasa digunakan:

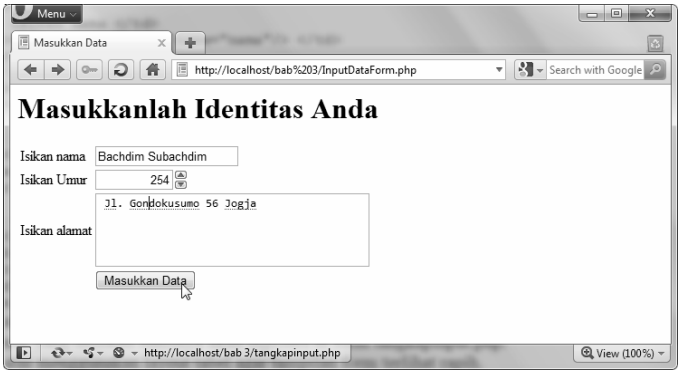
- GET: Variabel akan bisa dilihat di URL.
- POST: Variabel akan disembunyikan dan tidak terlihat di URL, lebih secure dan cocok untuk mem-parsing informasi yang penting, seperti password.

Selain itu Anda harus juga mengerti sintaks pembuatan form, yang terpenting adalah atribut “name” di masing-masing objek form (misalnya input atau text area). Atribut name ini penting karena nantinya digunakan untuk memanggil variabel di form lainnya.

Berikut kode untuk menerima masukan user. Kode ini bisa disimpan dalam format HTML atau PHP karena tidak ada kode PHP di dalamnya dan hanya berfungsi untuk me-render form dan menentukan action-nya.

```
<head>
 <title>Masukkan Data</title>
 </head>
<body>
<h1> Masukkanlah Identitas Anda</h1>
<form action="tangkapinput.php" method="post">
 <table> <!--untuk melayout-->
 <tr>
<td> Isikan nama </td>
 <td><input type="text" name="nama"/> </td>
</tr>
<td> Isikan Umur </td>
 <td><input type="number" name="umur"/> </td>
</tr>
<tr>
<td> Isikan alamat</td>
<td><textarea name="alamat" rows="5" cols="40">
 </textarea> </td>
 </tr>
<tr>
<td> <!--cuma untuk dummy layout--> </td>
<td><input type="submit" value="Masukkan Data" /></td></tr>
</table>
 </form>
</body>
</html>
```

Kode di atas mengandung kode action=”tangkapinput.php” yang menjelaskan bahwa kode yang diisikan nantinya akan diproses di form tangkapinput.php. Kode di atas menggunakan layout tabel agar tampilan form terlihat rapi.

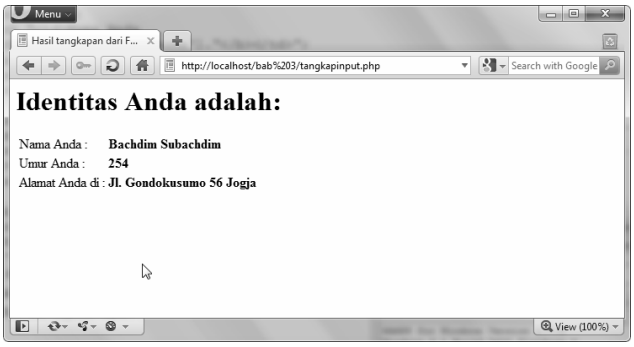


Gambar 3.13 Form untuk input data

Untuk memproses Anda perlu membuat terlebih dahulu file tangkapinput.php. Yang harus diperhatikan adalah nama objek-objek form yang akan digunakan untuk mengambil nilai variabel. Cara mengambilnya adalah dengan menggunakan sintaks $_POST['nama_variabel']." Misalnya untuk mengambil variabel yang diisikan di kotak dengan name = “nama”, maka digunakan sintaks $_POST['nama'].".

```
<html>
 <head>
 <title>Hasil tangkapan dari Form Terpisah</title>
 </head>
<body>
 <h1> Identitas Anda adalah:</h1>
 <table>
 <?php
 print "<tr>";
 print "<td>Nama Anda
:</td><td><b>".$_POST['nama']."</b></td>";
 print "</tr>";
 print "<tr>";
 print "<td>Umur Anda :</td><td><b>".$_POST['umur']."</b></td>
";
 print "</tr>";
 print "<tr>";
 print "<td>Alamat Anda di :
</td><td><b>".$_POST['alamat']."</b></td>";
 print "</tr>";
?>
</table>
 </body>
 </html>
```

Kode di atas akan mengambil variabel-variabel yang dimasukkan dari form di halaman sebelumnya yang kemudian ditampilkan dengan memanfaatkan label untuk me-layout-nya.



Gambar 3.14 Hasil tangkapan input dari form

3.7.2 Form dengan Input dan Pemrosesan Terpadu

Pada contoh di atas, Anda bisa melihat contoh pengolahan form dari PHP di halaman yang terpisah. Tapi Anda juga bisa membuat halaman pengolahan yang terletak satu halaman dengan form.

Untuk membuat form input dan halaman untuk menampilkan secara terpadu, Anda bisa menggunakan statement:

```
if (! Empty (nama_variabel))
```

Artinya, jika variabel yang dicari tidak kosong (alias ada) maka baru ditampilkan, sementara jika tidak ada, maka tidak akan ditampilkan. Hasilnya mirip dengan efek AJAX karena tulisan bisa langsung ditampilkan di satu halaman walaupun tidak persis seperti AJAX karena adanya refresh.

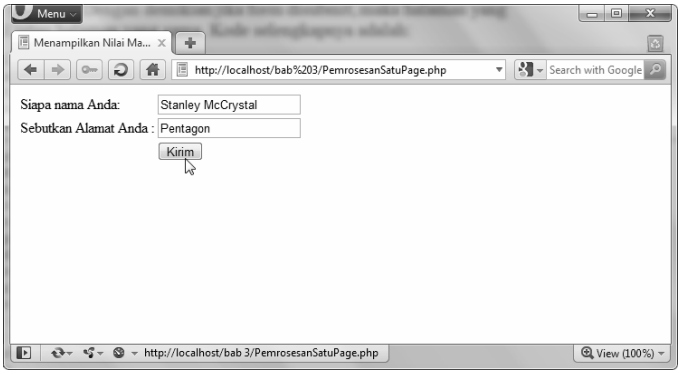
Yang perlu diketahui adalah digunakannya alamat action:

```
$_SERVER['PHP_SELF']
```

Artinya alamat action akan mengacu ke halaman itu sendiri dan bukan di halaman yang lain. Dengan demikian, jika form di-submit, maka halaman yang dipanggil tetap halaman yang sama. Kode selengkapnya adalah:

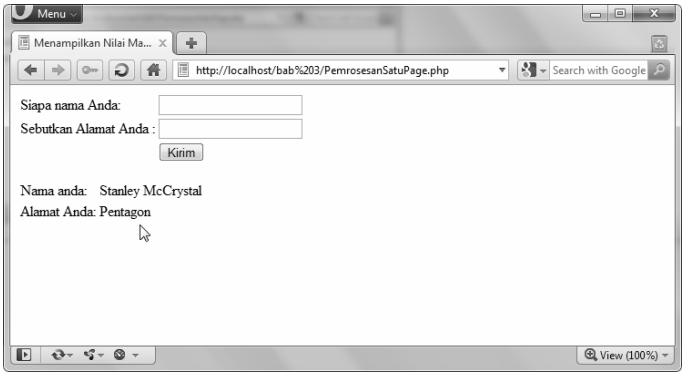
```
<html>
 <head>
 <title>Menampilkan Nilai Masukan dalam Satu Halaman</title>
 </head>
 <body>
 <form method="post" action="<?php print
$_SERVER['PHP_SELF']?>">
 <table>
 <tr>
 <td>
 Siapa nama Anda:
 </td>
 <td><input type="text" name="nama" /> </td>
 </tr>
 <tr>
<td> Sebutkan Alamat Anda :</td>
<td> <input type="text" name ="alamat"/> </td>
 </tr>
 <tr>
 <td> <!--ini dikosongkan untuk layout saja--></td>
 <td>
 <button type="submit"> Kirim </button>
 </td>
 </tr>
 </table>
 </form>
 <!--Tabel untuk menampilkan-->
 <table>
 <?php
 if ( ! empty( $_POST['nama'] ) ) {
     print "<tr><td>Nama anda:
</td><td>".$_POST['nama']."</td></tr>";
 }
  if ( ! empty( $_POST['alamat'] ) ) {
      print "<tr><td>Alamat Anda:
</td><td>".$_POST['alamat']."</td></tr>";
      }
 ?>
 </table>
 </body>
 </html>
```

Jika ditampilkan, muncul tampilan form dengan 2 kotak teks yang bisa Anda isikan teks, kemudian klik **Kirim** untuk mengirim.



Gambar 3.15 Klik pada Kirim untuk mengirim

Maka page akan me-refresh dan ada tampilan di dalamnya.



Gambar 3.16 Kode ditampilkan di bagian bawah form pada halaman yang sama

***